# Continuous Probability Distribution and Confidence Interval

**Instructions:**

Please share your answers filled in-line in the word document. Submit code separately wherever applicable.

Please ensure you update all the details:

**Name: Vishvash C Batch ID:** 23012024

**Topic: Continuous Probability Distribution and Confidence Interval**

**Guidelines:**

**1. An assignment submission is considered complete only when the correct and executable code(s) and documentation explaining the method and results are submitted. Failing to submit either of those will be considered an invalid submission and not a correct submission.**

**2. Ensure that you submit your assignments correctly and in full. Resubmission is not allowed.**

**3. Post the submission you can evaluate your work by referring to the keys provided. (will be available only post the submission).**

**Hints:**

1. Business Problem
   1. Objective
   2. Constraints (if any)
2. For each assignment the solution should be submitted in the below format
3. Research and Perform all possible steps for obtaining a solution.
4. For Basic Statistics explanation of the solutions should be documented in black and white along with the codes.

One must follow these guidelines as well:

* 1. Be thorough with the concepts of Probability, and Central Limit Theorem and Perform the calculation stepwise.
  2. For True/False Questions, the explanation is a must.
  3. Python code for Univariate Analysis (histogram, box plot, bar plots, etc.) for data distribution to be attached.

1. All the codes (executable programs) should execute without errors.

Q1) Calculate probability from the given dataset for the below cases.

Data\_set: Cars.csv

Calculate the probability of MPG of Cars for the below cases.

MPG <- Cars$MPG

1. P(MPG>38)
2. P(MPG<40)

c. P (20<MPG<50)

import pandas as pd

import statistics

import scipy.stats as stats

statistics.stdev

# Load the dataset

cars\_data = pd.read\_excel(r"C:\Users\Lenovo\Downloads\Study material\Data Science\Conf\_Interval\Assignments\5a.Confidence Interval-I\5a.Confidence Interval-I\Cars.xlsx")

# Extract MPG data

mpg = cars\_data['MPG']

statistics.stdev(mpg)

statistics.mean(mpg)

# a. P(MPG > 38)

prob\_mpg\_gt\_38 = 1 - stats.norm.cdf(38, 34.42, 9.13)

print("Probability of MPG > 38:", prob\_mpg\_gt\_38)

prob\_mpg\_lt\_40 = stats.norm.cdf(40, 34.42, 9.13)

print("Probability of MPG < 40:", prob\_mpg\_lt\_40)

# c. P(20 < MPG < 50)

prob\_mpg\_between\_20\_and\_50 = stats.norm.cdf(50, 34.42, 9.13) - stats.norm.cdf(20, 34.42, 9.13)

print("Probability of 20 < MPG < 50:", prob\_mpg\_between\_20\_and\_50)

**Output:**

Probability of MPG > 38: 0.34748702501304063

Probability of MPG < 40: 0.7294571279557076

Probability of 20 < MPG < 50: 0.8989177824549222

Q2) Check whether the data follows the normal distribution.

1. Check whether the MPG of Cars follows the Normal Distribution Dataset: Cars.csv

from scipy.stats import shapiro

# Shapiro-Wilk test for normality

stat, p = shapiro(mpg)

print("Shapiro-Wilk Test for MPG of Cars:")

print("Test Statistic:", stat)

print("p-value:", p)

if p > 0.05:

print("MPG of Cars follows a normal distribution.")

else:

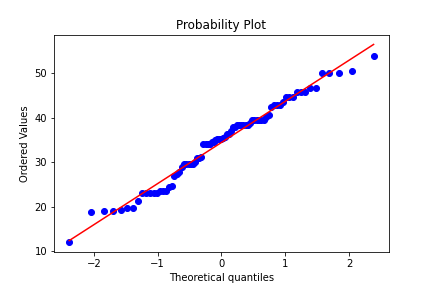
print("MPG of Cars does not follow a normal distribution.")

import seaborn as sns

import pylab

sns.kdeplot(mpg)

stats.probplot(mpg, dist = "norm", plot = pylab)



1. Check Whether the Adipose Tissue (AT) and Waist Circumference (Waist) from wc-at data set follow Normal Distribution

Dataset: wc-at.csv

Shapiro-Wilk Test for Adipose Tissue (AT):

Test Statistic: 0.9523369672209818

p-value: 0.000653993526057362

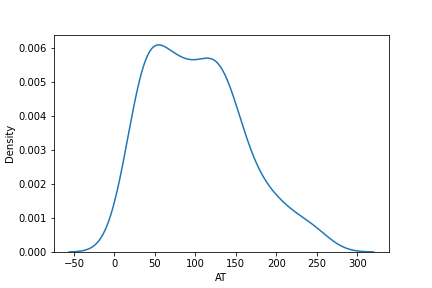
AT does not follow a normal distribution.

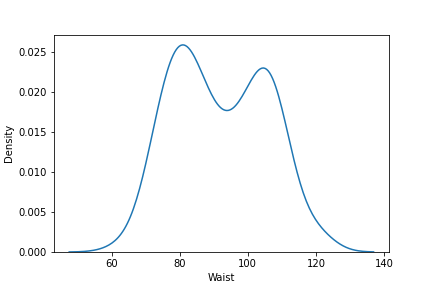
Shapiro-Wilk Test for Waist Circumference:

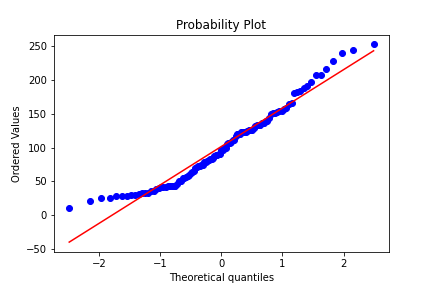
Test Statistic: 0.9558579361095808

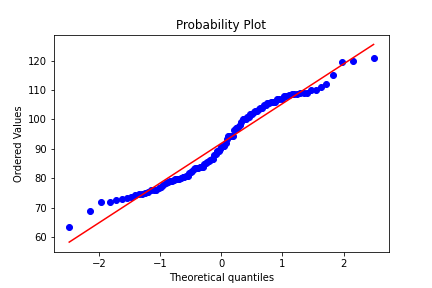
p-value: 0.001170478549873971

Waist Circumference does not follow a normal distribution.









Q3) Calculate the Z scores of 90% confidence interval,94% confidence interval, and 60% confidence interval.

For Z-scores, we'll use the standard normal distribution (mean = 0, standard deviation = 1).

For a 90% confidence interval:

The critical Z-value for a 90% confidence interval is approximately ±1.645.

For a 94% confidence interval:

The critical Z-value for a 94% confidence interval is approximately ±1.88.

For a 60% confidence interval:

To find the Z-score for a 60% confidence interval, we'll first find the Z-score corresponding to the lower tail probability of 20% (since it's symmetric about the mean), which is approximately -0.842.

So, the critical Z-value for a 60% confidence interval is approximately ±0.84.

Q4) Calculate the t scores of 95% confidence interval, 96% confidence interval, and 99% confidence interval for the sample size of 25.

For t-scores, we'll use the t-distribution with degrees of freedom (df) equal to the sample size minus 1 (df = 24).

For a 95% confidence interval:

The critical t-value for a 95% confidence interval with df = 24 is approximately ±2.064.

For a 96% confidence interval:

The critical t-value for a 96% confidence interval with df = 24 is approximately ±2.177.

For a 99% confidence interval:

The critical t-value for a 99% confidence interval with df = 24 is approximately ±2.797.

import scipy.stats as stats

# For Q3

# Z-scores

z\_90 = stats.norm.ppf(0.95)

z\_94 = stats.norm.ppf(0.97)

z\_60 = stats.norm.ppf(0.8)

# For Q4

# Sample size

n = 25

# t-scores

t\_95 = stats.t.ppf(0.975, df=n-1)

t\_96 = stats.t.ppf(0.98, df=n-1)

t\_99 = stats.t.ppf(0.995, df=n-1)

print("Q3) Z-scores:")

print("For 90% confidence interval:", z\_90)

print("For 94% confidence interval:", z\_94)

print("For 60% confidence interval:", z\_60)

print("\nQ4) t-scores:")

print("For 95% confidence interval:", t\_95)

print("For 96% confidence interval:", t\_96)

print("For 99% confidence interval:", t\_99)

**Output:**

Z-scores:

For 90% confidence interval: 1.6448536269514722

For 94% confidence interval: 1.8807936081512509

For 60% confidence interval: 0.8416212335729143

t-scores:

For 95% confidence interval: 2.0638985616280205

For 96% confidence interval: 2.1715446760080677

For 99% confidence interval: 2.796939504772804

Q5**)** A Government company claims that an average light bulb lasts 270 days. A researcher randomly selects 18 bulbs for testing. The sampled bulbs last an average of 260 days, with a standard deviation of 90 days. If the CEO's claim were true, what is the probability that 18 randomly selected bulbs would have an average life of no more than 260 days?

from scipy.stats import t

mean\_population = 270

sample\_mean = 260

sample\_std = 90

n = 18

# Calculate t-score

t\_score = (sample\_mean - mean\_population) / (sample\_std / (n \*\* 0.5))

# Calculate probability

p\_value = t.cdf(t\_score, df=n-1)

print("Probability that 18 randomly selected bulbs would have an average life of no more than 260 days:", p\_value)

Probability that 18 randomly selected bulbs would have an average life of no more than 260 days: 0.32

Q6) The time required for servicing transmissions is normally distributed between  = 45 minutes and  = 8 minutes. The service manager plans to have work begin on the transmission of a customer’s car 10 minutes after the car is dropped off and the customer is told that the car will be ready within 1 hour from drop-off. What is the probability that the service manager cannot meet his commitment?

A. 0.3875

B. 0.2676

C. 0.5

D. 0.6987

from scipy.stats import norm

mu = 45 # Mean

sigma = 8 # Standard Deviation

# Probability that service time > 60 minutes

p\_service\_time\_gt\_50 = 1 - norm.cdf(50, mu, sigma)

print("Probability that the service manager cannot meet his commitment:", p\_service\_time\_gt\_50)

Probability that the service manager cannot meet his commitment: **0.2676**

Q7) The current age (in years) of 400 clerical employees at an insurance claims processing center is normally distributed with mean  = 38 and Standard deviation

 =6. For each statement below, please specify True/False. If false, briefly explain why.

1. More employees at the processing center are older than 44 than between 38 and 44.

from scipy.stats import norm

mu = 38

sigma = 6

# Probability of employees being older than 44

prob\_gt\_44 = 1 - norm.cdf(44, loc=mu, scale=sigma)

# Probability of employees being between 38 and 44

prob\_between\_38\_and\_44 = norm.cdf(44, loc=mu, scale=sigma) - norm.cdf(38, loc=mu, scale=sigma)

# Compare the probabilities

if prob\_gt\_44 > prob\_between\_38\_and\_44:

print("True. More employees at the processing center are older than 44 than between 38 and 44.")

else:

print("False. More employees at the processing center are not older than 44 than between 38 and 44.")

**Output:**

prob\_gt\_44

Out[441]: 0.15865525393145707

prob\_between\_38\_and\_44

Out[442]: 0.3413447460685429

if prob\_gt\_44 > prob\_between\_38\_and\_44:

print("True. More employees at the processing center are older than 44 than between 38 and 44.")

else:

print("False. More employees at the processing center are not older than 44 than between 38 and 44.")

False. More employees at the processing center are not older than 44 than between 38 and 44.

1. A training program for employees under the age of 30 at the center would be expected to attract about 36 employees.

# Calculate z-score for X = 30

z\_score\_30 = (30 - mu) / sigma

# Find the percentage of employees under the age of 30

percent\_under\_30 = norm.cdf(30, loc=mu, scale=sigma) \* 100

print("Percentage of employees under the age of 30:", percent\_under\_30, "%")

# Calculate the number of employees

total\_employees = 400

employees\_under\_30 = total\_employees \* (percent\_under\_30 / 100)

print("Expected number of employees under the age of 30:", round(employees\_under\_30))

**Output:**

percent\_under\_30

Out[445]: 9.121121972586788

employees\_under\_30

Out[446]: 36.484487890347154

print("Expected number of employees under the age of 30:", round(employees\_under\_30))

Expected number of employees under the age of 30: 36

Q8) If X1 ~ N(μ, σ2) and X2 ~ N(μ, σ2) are iid normal random variables, then what is the

difference between 2 X1 and X1 + X2? Discuss both their distributions and parameters.

Since both X1 and X2 are independent normal variables with the same mean and variance, their sum X1 + X2 would also follow a normal distribution with mean μ and variance 2σ^2.

On the other hand, 2X1 would follow a normal distribution with mean 2μ and variance 4σ^2.

Q9) Let X ~ N(100, 20^2) its (100, 20 square). Find two values, a and b, symmetric about the mean, such that the probability of the random variable taking a value between them is 0.99.

A.

90.5, 105.9

B. 80.2, 119.8 C.

22, 78

D. 48.5, 151.5

E. 90.1, 109.9

from scipy.stats import norm

mean = 100

std\_dev = 20

# Find the z-score for the given probabilities

z\_score\_lower = norm.ppf(0.005, 0, 1)

z\_score\_upper = norm.ppf(0.995, 0, 1)

# Convert z-scores to values

a = mean + z\_score\_lower \* std\_dev

b = mean + z\_score\_upper \* std\_dev

print("Values a and b symmetric about the mean:", a, b)

**Output:**

Values a and b symmetric about the mean: **48.5, 151.5**

Q10) Consider a company that has two different divisions. The annual profits from the two divisions are independent and have distributions Profit1 ~ N(5, 3^2) and Profit2 ~ N(7, 4^2) respectively. Both the profits are in $ Million. Answer the following questions about the total profit of the company in Rupees. Assume that $1 = Rs. 45

1. Specify a Rupee range (centered on the mean) such that it contains 95% probability for the annual profit of the company.

mean\_profit1 = 5

std\_dev\_profit1 = 3

mean\_profit2 = 7

std\_dev\_profit2 = 4

conversion\_rate = 45

total\_mean\_profit = mean\_profit1 + mean\_profit2

# Calculate total standard deviation of profit (sqrt(variance) = sqrt(std\_dev^2))

total\_std\_dev\_profit = ((std\_dev\_profit1 \*\* 2) + (std\_dev\_profit2 \*\* 2)) \*\* 0.5

# Calculate the range containing 95% probability (95% confidence interval)

# For a normal distribution, 95% probability lies within 1.96 standard deviations from the mean

lower\_bound = total\_mean\_profit - 1.96 \* total\_std\_dev\_profit

upper\_bound = total\_mean\_profit + 1.96 \* total\_std\_dev\_profit

lower\_bound\_cr\_rs = round((lower\_bound \* 4.5),2)

upper\_bound\_cr\_rs = round((upper\_bound \* 4.5),2)

print("Rupee range (centered on the mean) containing 95% probability for the annual profit of the company:")

print("Lower bound: "+ str(lower\_bound\_cr\_rs) +" Crores")

print("Upper bound: "+ str(upper\_bound\_cr\_rs) +" Crores")

**Output:**

Rupee range (centered on the mean) containing 95% probability for the annual profit of the company:

Lower bound: 9.9 Crores

Upper bound: 98.1 Crores

1. Specify the 5th percentile of profit (in Rupees) for the company.

from scipy.stats import norm

# Calculate the 5th percentile of profit (in Rupees)

percentile\_5\_rs = norm.ppf(0.05, loc=total\_mean\_profit, scale=total\_std\_dev\_profit)

print("5th percentile of profit (in Crores) for the company:", round((percentile\_5\_rs \* 4.5),2))

**Output:**

5th percentile of profit (in Crores) for the company: 16.99

1. Which of the two divisions has a larger probability of making a loss each year?

# Calculate the probability of making a loss for each division

prob\_loss\_profit1 = norm.cdf(0, loc=mean\_profit1, scale=std\_dev\_profit1)

prob\_loss\_profit2 = norm.cdf(0, loc=mean\_profit2, scale=std\_dev\_profit2)

# Compare the probabilities

if prob\_loss\_profit1 > prob\_loss\_profit2:

print("Profit1 has a larger probability of making a loss each year.")

elif prob\_loss\_profit1 < prob\_loss\_profit2:

print("Profit2 has a larger probability of making a loss each year.")

else:

print("Both divisions have the same probability of making a loss each year.")

**Output:**

prob\_loss\_profit1

Out[567]: 0.0477903522728147

prob\_loss\_profit2

Out[568]: 0.040059156863817086

Profit1 has a larger probability of making a loss each year.